OBJECT ORIENTED PROGRAMMING WITH JAVA 8– LAB 9

1. Write a program to find if a given string is a palindrome or not using StringBuilder.

Ans=

Code-

```
import java.util.Scanner;

public class Palindrome
{
        public static boolean CheckPalindrome(String S)
        {
         StringBuilder RS = new StringBuilder(S).reverse();
         String reversed = RS.toString();
         return S.equalsIgnoreCase(reversed);
        }

        public static void main(String[] args)
        {
         Scanner P = new Scanner(System.in);
         System.out.print("Enter a string: ");
         String input = P.nextLine();

         if(CheckPalindrome(input))
                {
                 System.out.println("Entered string is Palindrome.");
                }
         else
                {
                 System.out.println("Entered string is not Palindrom.");
                }
        }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>javac Palindrome.java

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java Palindrome
Enter a string: Hello
Entered string is not Palindrom.

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java Palindrome
Enter a string: level
Entered string is Palindrome.

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>
```

2. Accept a line of text. Find the reverse of each word and display the string.

Ans=

Code-

```
import java.util.Scanner;

public class Reverse
{
        public static String reverseText(String input)
        {
         String[] words = input.split(" ");
         StringBuilder reversedString = new StringBuilder();

         for (String word : words)
        {
          StringBuilder reversedWord = new StringBuilder(word).reverse();
          reversedString.append(reversedWord).append(" ");
        }
        return reversedString.toString().trim();
        }

        public static void main(String[] args)
        {
         Scanner P = new Scanner(System.in);

         System.out.print("Enter a line of text: ");
         String inputString = P.nextLine();

         String reversedWordsString = reverseText(inputString);
         System.out.println("Reversed words string: " + reversedWordsString);
        }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>javac Reverse.java

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java Reverse
Enter a line of text: Was it a car or cat I saw
Reversed words string: saW ti a rac ro tac I was

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>
```

3. A sentence is terminated by either “.”, “!”, “?”. Input a piece of text containing sentences. Obtain the length of the sentence and frequency of vowels in each sentence.

Ans=

Code-

```
import java.util.Scanner;

public class Sentence
{
         public static int calculateSentenceLength(String sentence)
        {
         String[] w = sentence.trim().split("\\s+");
         return w.length;
        }

        public static int calculateVowelFrequency(String sentence)
        {
         String lowercaseSentence = sentence.toLowerCase();
         String vowels = "aeiou";
         int vowelCount = 0;

         for (char c : lowercaseSentence.toCharArray())
                {
                 if (vowels.indexOf(c) != -1)
                        {
                         vowelCount++;
                        }
                }
        return vowelCount;
        }
```

```
        public static void main(String[] args)
        {
         Scanner P = new Scanner(System.in);

         System.out.print("Enter a sentences: ");
         String inputText = P.nextLine();
         String[] sentences = inputText.split("[.!?]");

         for (String sentence : sentences)
                {
                 int sentenceLength = calculateSentenceLength(sentence);
                 int vFrequency = calculateVowelFrequency(sentence);

                 System.out.println("Sentence: " + sentence);
                 System.out.println("Sentence Length: " + sentenceLength);
                 System.out.println("Vowel Frequency: " + vFrequency);
                 System.out.println();
                }
        }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>javac Sentence.java

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java Sentence
Enter a sentences: Hello! How are you? Today is good day.
Sentence: Hello
Sentence Length: 1
Vowel Frequency: 2

Sentence:  How are you
Sentence Length: 3
Vowel Frequency: 5

Sentence:  Today is good day
Sentence Length: 4
Vowel Frequency: 6


C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>
```

4. Create a functional interface named Verify with one abstract method boolean check(int a). Create a class that implements this interface using lambda expression where the method returns true if the number is prime and false otherwise.

Ans=

Code- Part1,

```
import java.util.Scanner;

@FunctionalInterface
interface Verify
{
        boolean check(int a);
}

public class VerifyPrime
{
        public static void main(String[] args)
        {
         Scanner P = new Scanner(System.in);
         System.out.print("Enter a number: ");
         int number = P.nextInt();

         Verify Prime = (int num) ->
                {
                 if(num <= 1)
                        {
                         return false;
                        }
                 if (num == 2)
                        {
                         return true;
                        }
                if (num % 2 == 0)
                        {
                         return false;
                        }
```

Part2,

```
        int sqrt = (int) Math.sqrt(num);
                for (int i = 3; i <= sqrt; i += 2)
                {
                 if (num % i == 0)
                        {
                         return false;
                        }
                }
         return true;
        };

        boolean PrimeResult = Prime.check(number);

        if (PrimeResult)
                {
                 System.out.println(number + " is a prime number.");
                }
        else    {
                 System.out.println(number + " is not a prime number.");
                }
        }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>javac VerifyPrime.java

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java VerifyPrime
Enter a number: 8
8 is not a prime number.

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java VerifyPrime
Enter a number: 7
7 is a prime number.

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>
```

5. Create a functional interface named Concatenation with one abstract method String join(String a, String b). Write a program to implement lambda expression to concatenate two strings.

Ans=

Code-

```
import java.util.Scanner;

@FunctionalInterface
interface Concatenation
{
        String join(String a, String b);
}

public class StringConcatenation
{
         public static void main(String[] args)
         {
          Scanner P= new Scanner(System.in);

          System.out.print("Enter the first string: ");
          String s1 = P.nextLine();

          System.out.print("Enter the second string: ");
          String s2 = P.nextLine();

          Concatenation c = (a, b) -> a + b;

          String result = c.join(s1, s2);

          System.out.println("Concatenated String: " + result);
    }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>javac StringConcatenation.java

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java StringConcatenation
Enter the first string: Hello!
Enter the second string: World.
Concatenated String: Hello!World.

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>
```